Configuration and environment setup of light environment template

1. The template source'ssetup related to the web environment is defined and used at the following 3 locations.
 - web.xml
 - classpath*:egovframework/spring/com/context-*.xml
 - /WEB-INF/config/egovframework/springmvc/*.xml
2. DB related setup of template source is defined and used at globlas.properties.

Main setup of web.xml

- In the case of being called as *.do, cross site script by special character shall be prevented by setting up the encoding, which is outputted to basic screen, so that it can be filtered with utf-8. If HTMLTagFilter is set up, the method of converting the value itself which is conveyed to parameter can be used. In template source, the method of processing encoding at the moment of output is basically used without converting the value inputted through screen.

```
<filter>
<filter-name>encodingFilter</filter-name>
        <filter-class>
                org.springframework.web.filter.CharacterEncodingFilter
        </filter-class>
        <init-param>
                <param-name>encoding</param-name>
                <param-value>utf-8</param-value>
        </init-param>
        </filter>
        <filter>
<filter-name>HTMLTagFilter</filter-name>
<filter-class>
egovframework.rte.ptl.mvc.filter.HTMLTagFilter
</filter-class>
</filter>
<filter-mapping>
        <filter-name>encodingFilter</filter-name>
        <url-pattern>*.do</url-pattern>
</filter-mapping>
<!-- For template, basic policy is to use UTF-8 encoding filter and <c:out> tag during JSP output. -->
<!-- Following HtmlTagFilter can also be used through additional registration and reinforcement of function. -->
<!--
<filter-mapping>
<filter-name>HTMLTagFilter</filter-name>
<url-pattern>*.do</url-pattern>
</filter-mapping>
    -->
```

- Register the location of the spring context related environment setup information file.

```
<context-param>
                <param-name>contextConfigLocation</param-name>
                <param-value>
                        classpath*:egovframework/spring/com/context-*.xml
                </param-value>
        </context-param>
```

- Register the setup of Listener, DispatcherServlet for spring mvc and location of mvc related additional environment setup information file.

```
<listener>
```

```
		<listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>
	</listener>

	<servlet>
		<servlet-name>action</servlet-name>
		<servlet-class>
			org.springframework.web.servlet.DispatcherServlet
		</servlet-class>
		<init-param>
			<param-name>contextConfigLocation</param-name>
			<param-value>
				/WEB-INF/config/egovframework/springmvc/*.xml
			</param-value>
		</init-param>
		<load-on-startup>1</load-on-startup>
	</servlet>
	<servlet-mapping>
		<servlet-name>action</servlet-name>
		<url-pattern>*.do</url-pattern>
	</servlet-mapping>
```

- The setup which is included when the spring security is used (Only internal business and portal site template are included.)

```
<filter>
<filter-name>springSecurityFilterChain</filter-name>
<filter-class>org.springframework.web.filter.DelegatingFilterProxy</filter-class>
</filter>
<filter-mapping>
<filter-name>springSecurityFilterChain</filter-name>
<url-pattern>/*</url-pattern>
</filter-mapping>

<listener>
<listener-class>org.springframework.security.ui.session.HttpSessionEventPublisher</listener-class>
</listener>
```

Main setup of egovframework/spring/com/context-*.xml

Additional setup files are located under each of the resource/egovframework/spring/com/folders of template project source.
The setup information related with anontation, aop, schedule, validation, security, id-generation, sqlmap, and transaction are registered. (There is difference for each type of template.)

- ExceptionTransfer setup of context-aspect.xml (In the case of being occurred at template service impl class, individual log for Exception is outputted.)

```
<aop:config>
<aop:pointcut id="serviceMethod"
expression="execution(* egovframework.let..impl.*Impl.*(..)) or execution(*
egovframework.com..impl.*Impl.*(..))" />

<aop:aspect ref="exceptionTransfer">
<aop:after-throwing throwing="exception"
pointcut-ref="serviceMethod" method="transfer" />
</aop:aspect>
</aop:config>
    ...
```

- egovMessageSource setup of context-common.xml (utilizing the message registered at property file), setup of leavaTrace (When Exception is generated, individual trace is outputted.)

```
<bean id="egovMessageSource" class="egovframework.com.cmm.EgovMessageSource">
<property name="reloadableResourceBundleMessageSource">
<ref bean="messageSource" />
</property>
</bean>
    ...

<!-- It shall be set up so that LeveaTrace of execution environment can be utilized to conduct separate work for post-processing when the Exception is generated. -->
<bean id="leaveaTrace" class="egovframework.rte.fdl.cmmn.trace.LeaveaTrace">
        <property name="traceHandlerServices">
                <list>
                        <ref bean="traceHandlerService" />
                </list>
        </property>
</bean>
    ...

<!-- Application example of egovMessageSource within source
@Resource(name="egovMessageSource")
EgovMessageSourceegovMessageSource;
        String message = egovMessageSource.getMessage("fail.common.login");
    -->
<!-- Application example within the source (In template, there isn't any special post-processing, therefore, only the sample source is applied during logout.)
@Resource(name="leaveaTrace") LeaveaTraceleaveaTrace;
    ...
    } catch (ArithmeticExceptionathex) {
        // Post-processing etc. in accordance with message can be conducted at TraceHandler.
leaveaTrace.trace("fail.common.msg", this.getClass());
    }
    ...
    -->
```

- datasource setup of context-datasource.xml

```
<!-- Define the datasource information with the DB information registered at globals.properties files. -->
<bean id="propertyConfigurer"
class="org.springframework.beans.factory.config.PropertyPlaceholderConfigurer">
<property name="locations">
<list>
<value>classpath:/egovframework/egovProps/globals.properties</value>
</list>
</property>
</bean>

<!-- Setup of datasource (utilization of propertyConfigurer) -->
<alias name="dataSource-${Globals.DbType}" alias="dataSource" />
<alias name="dataSource-${Globals.DbType}" alias="egov.dataSource" />

<!--mysql -->
<bean id="dataSource-mysql" class="org.apache.commons.dbcp.BasicDataSource" destroy-method="close">
<property name="driverClassName" value="${Globals.DriverClassName}"/>
<property name="url" value="${Globals.Url}" />
<property name="username" value="${Globals.UserName}"/>
<property name="password" value="${Globals.Password}"/>
</bean>
    ...

<!-- Definition sample for 4 types of DB is existed. -->
```

```
<!-- After changing dbtype of home page template globals.properties to hsql, simple test can be performed with the following setup.  -->
<bean id="dataSource-hsql" class="org.apache.commons.dbcp.BasicDataSource" destroy-method="close">
        <property name="driverClassName" value="net.sf.log4jdbc.DriverSpy"/>
        <property name="url" value="jdbc:log4jdbc:hsqldb:hsql://localhost/sampledb"/>
        <property name="username" value="sa"/>
</bean>
```

- Setup of serial number generation service for each business of context-idgen.xml

```
<!-- Setup of bean class which creates serial number when record is newly generated for each business at system -->
<!— Attached file ID Generation  Config-->
<bean name="egovFileIdGnrService"
class="egovframework.rte.fdl.idgnr.impl.EgovTableIdGnrService"
destroy-method="destroy">
<property name="dataSource" ref="dataSource-${Globals.DbType}" />
<property name="strategy" ref="fileStrategy" />
<property name="blockSize"  value="10"/>
<property name="table"       value="IDS"/>
<property name="tableName"  value="FILE_ID"/>
</bean>
<!-- Attached file ID Generation  Strategy Config -->
<bean name="fileStrategy"
class="egovframework.rte.fdl.idgnr.impl.strategy.EgovIdGnrStrategyImpl">
<property name="prefix" value="FILE_" />
<property name="cipers" value="15" />
<property name="fillChar" value="0" />
</bean>
    ...

<!-- Application example within the source
       ...
@Resource(name = "egovFileIdGnrService") private EgovIdGnrServiceidgenService;
    ...
    String newId = idgenService.getNextStringId();
    ...
    -->
```

- Property value setup of context-properties.xml

```
<!-- Setup of property information -->
<bean name="propertiesService"
class="egovframework.rte.fdl.property.impl.EgovPropertyServiceImpl" destroy-method="destroy">
<property name="properties">
<map>
<entry key="pageUnit" value="10"/>
<entry key="pageSize" value="10"/>
<entry key="posblAtchFileSize" value="5242880"/>
<entry key="Globals.fileStorePath" value="/user/file/sht/"/>
<entry key="Globals.addedOptions" value="false"/>
</map>
</property>
</bean>

<!-- Application example within the source
@Resource(name = "propertiesService") protected EgovPropertyServicepService;
    ...
    String savePath = pService.getString("Globals.fileStorePath");
    ...
    -->
```

- Property value setup of context-sqlMap.xml

```
<!-- With the bean setup by confirming the location information of the query to be used at DAO within the source, it is used by being included inside of AbstractDao which extends at Dao class -->
<!-- Register query location information files at the items of configLocations properties. (In each of location information files, the location information of files in which actual queries are included is existed.) -->
<!--SqlMap setup for iBATIS Database Layer -->
<bean id="sqlMapClient" class="org.springframework.orm.ibatis.SqlMapClientFactoryBean">
<property name="configLocations">
<list>
<value>classpath:/egovframework/sqlmap/config/${Globals.DbType}/*.xml</value>
</list>
</property>
<property name="dataSource" ref="dataSource-${Globals.DbType}"/>
<property name="lobHandler" ref="lobHandler"/>
</bean>
    ...

<!-- Application example within the source (It is automatically applied through extends when defining the DAO class) -->
<!-- It can also be applied by re-defining EgovAbstractDAO after newly designated as in the case of com.sqlMapClient. (Refer to the setup within the source.) -->
@Repository("BBSAttributeManageDAO")
public class BBSAttributeManageDAO extends EgovAbstractDAO {
    ...
```

* Transaction setup of context-transaction.xml

```
<bean id="txManager" class="org.springframework.jdbc.datasource.DataSourceTransactionManager">
        <property name="dataSource" ref="dataSource"/>
        </bean>

        <tx:advice id="txAdvice" transaction-manager="txManager">
                <tx:attributes>
                        <tx:method name="*" rollback-for="Exception"/>
                </tx:attributes>
        </tx:advice>

        <aop:config>
                <aop:pointcut id="requiredTx"
                        expression="execution(* egovframework.let..impl.*Impl.*(..)) or execution(* egovframework.com..*Impl.*(..))"/>
                <aop:advisor advice-ref="txAdvice"
                        pointcut-ref="requiredTx" />
        </aop:config>

<!-- From template, transaction is processed within the function of service impl class. (If exception is generated, all the processing results of DAO functions which have been called within the function of impl class are rollbacked.) -->
```

* Vaildator setup of context-validator.xml

```
<bean id="beanValidator" class="org.springmodules.validation.commons.DefaultBeanValidator">
<property name="validatorFactory" ref="validatorFactory"/>
</bean>

<bean id="validatorFactory" class="org.springmodules.validation.commons.DefaultValidatorFactory">
<property name="validationConfigLocations">
<list>
<!-- Setup of light environment template validator -->
        <value>classpath:/egovframework/validator/validator-rules-let.xml</value>
```

```
        <value>classpath:/egovframework/validator/let/**/*.xml</value>
</list>
</property>
</bean>

<!-- When validation is conducted in the xml file which is registered as the property of
validationConfigLocations, the function to be used and the form information which is the subject of
validation are registered.-->
<!-- Example of classpath:/egovframework/validator/validator-rules-let.xml contents
        ...
<validator name="required"
        ...
<javascript><![CDATA[
functionvalidateRequired(form) {
varisValid = true;
                ...
    -->
<!-- Example of classpath:/egovframework/validator/let/*.xml contents
        ...
<validator name="required"
        ...
<javascript><![CDATA[
functionvalidateRequired(form) {
varisValid = true;
                ...
    -->
<!--
<formset>
                    <form name="boardMaster">
                            <field property="bbsNm" depends="required">
                    <arg0 key="cop.bbsNm" />
                    </field>
            ...
    -->
```

* Spring security setup of context-security.xml (Internal business template and portal site template are corresponded to this.)

```
<!-- This can be used by adjusting the authority control setup in which jdbcService is used with the
setup for spring security related beans. -->
<b:bean id="jdbcUserService"
class="egovframework.let.sec.security.userdetails.jdbc.EgovJdbcUserDetailsManager" >
<b:property name="usersByUsernameQuery" value="SELECT USER_ID, ESNTL_ID AS PASSWORD, 1
ENABLED, USER_NM, USER_ZIP, USER_ADRES, USER_EMAIL, USER_SE, usr.ORGNZT_ID, ESNTL_ID,
org.ORGNZT_NM FROM COMVNUSERMASTER usr, LETTNORGNZTINFO org WHERE usr.ORGNZT_ID =
org.ORGNZT_ID AND CONCAT(USER_SE, USER_ID) = ? "/>

<b:property name="authoritiesByUsernameQuery" value="SELECT A.SCRTY_DTRMN_TRGET_ID
USER_ID, A.AUTHOR_CODE AUTHORITY FROM LETTNEMPLYRSCRTYESTBS A, COMVNUSERMASTER B
WHERE A.SCRTY_DTRMN_TRGET_ID = B.ESNTL_ID AND B.USER_ID = ? "/>
<b:property name="roleHierarchy" ref="roleHierarchy"/>
<b:property name="dataSource" ref="dataSource"/>
<b:property name="mapClass"
value="egovframework.let.sec.security.common.EgovSessionMapping"/>
</b:bean>
```

* The excel file control setup of context-excel.xml (This is corresponded to internal business template.)

```
<bean id="excelZipService"          class="egovframework.rte.fdl.excel.impl.EgovExcelServiceImpl">
<property name="propertyPath" value="excelInfo.xml" />
<property name="mapClass"
value="egovframework.let.sym.ccm.zip.service.impl.EgovCcmExcelZipMapping" />
```

```
<property name="sqlMapClient" ref="sqlMapClient" />
</bean>
<!-- Application example within the source (After setting up excel column information and related information (Refer to the EgovCcmExcelZipMapping class of template), the uploadExcel function is used during the excel upload.)
            ...
excelZipService.uploadExcel("ZipManageDAO.insertExcelZip", file, 2, (long) 5000);
      ...
    -->
```

* Setup of daily connection statistics scheduling of context-scheduling.xml (This is corresponded to internal business template.)//

```
<!-- Summary of system log   -->
<bean id="sysLogging"
class="org.springframework.scheduling.quartz.MethodInvokingJobDetailFactoryBean">
<property name="targetObject" ref="egovSysLogScheduling" />
<property name="targetMethod" value="sysLogSummary" />
<property name="concurrent" value="false" />
</bean>
<bean id="sysLogTrigger" class="org.springframework.scheduling.quartz.SimpleTriggerBean">
<property name="jobDetail" ref="sysLogging" />
<!-- Execute at 1 minute after the starting. (milisecond) 60000-->
<property name="startDelay" value="30000" />
<!-- Execute every 24 hours.864000000 (milisecond) -->
<property name="repeatInterval" value="864000000" />
</bean>
<bean id="sysLogScheduler"
class="org.springframework.scheduling.quartz.SchedulerFactoryBean">
<property name="triggers">
<list><ref bean="sysLogTrigger" /></list>
</property>
</bean>
```

<!-- Execute in daily unit of sysLogSummary() of egovSysLogScheduling service based on the time of 1 minute after the server operation -->

* Setup of the log record which uses aop of context-syslogaop.xml (This is corresponded to internal business template.)

```
<bean id="logManage" class="egovframework.let.sym.log.clg.service.EgovLoginLogAspect" />
<aop:config>
<aop:aspect id="logManageAspect" ref="logManage">
<!-- Login Method -->
<aop:beforepointcut="execution(public *
egovframework.let.uat.uia.service.EgovLoginService.actionLogin(..))" method="logLogin" />
          </aop:aspect>
</aop:config>
```

<!-- When the function started with actionLogin is operated in the login related service class, connection log is registered through logManage. -->

/WEB-INF/config/egovframework/springmvc/*.xml

Additional setup files are located under each of the WEB-INF/config/egovframework/springmvc folders of template project source.
Related setup information of exception, view, error-page is registered.

- Setup of mvc for auto detect

```
<context:component-scan base-package="egovframework">
<context:include-filter type="annotation" expression="org.springframework.stereotype.Controller"/>
```

```
<context:exclude-filter type="annotation" expression="org.springframework.stereotype.Service"/>
<context:exclude-filter type="annotation" expression="org.springframework.stereotype.Repository"/>
</context:component-scan>
```

- Setup for conveying the argument in the shape of anotation request mapping and commandmap

```
<bean class="org.springframework.web.servlet.mvc.annotation.AnnotationMethodHandlerAdapter">
<property name="webBindingInitializer">
<bean class="egovframework.com.cmm.web.EgovBindingInitializer"/>
</property>
<property name="customArgumentResolvers">
<list>
<bean class="egovframework.rte.ptl.mvc.bind.CommandMapArgumentResolver"/>
</list>
</property>
</bean>
<bean class="org.springframework.web.servlet.mvc.annotation.DefaultAnnotationHandlerMapping"/>
```

- Setup of error page which is to be moved when exception is generated

```
<bean class="org.springframework.web.servlet.handler.SimpleMappingExceptionResolver">
<property name="defaultErrorView" value="cmm/error/egovError"/>
<property name="exceptionMappings">
<props>
<prop key="org.springframework.dao.DataAccessException">cmm/error/dataAccessFailure</prop>
<prop
key="org.springframework.transaction.TransactionException">cmm/error/transactionFailure</prop>
<prop key="egovframework.rte.fdl.cmmn.exception.EgovBizException">cmm/error/egovError</prop>
<prop key="org.springframework.security.AccessDeniedException">cmm/error/accessDenied</prop>
</props>
</property>
</bean>
```

- Setup for prefix, suffix handling of JSP file name for screen processing

```
<bean class="org.springframework.web.servlet.view.UrlBasedViewResolver" p:order="1"
p:viewClass="org.springframework.web.servlet.view.JstlView"
p:prefix="/WEB-INF/jsp/" p:suffix=".jsp"/>
```

- Setup of filter which checks URL for which login is required (Only home page template is used, and for other templates, security authority is applied.)

```
<!-- In the home page template, together with filter setup, restriction for URL allowed for the login
administrator is checked through access check function at controller class.-->
<!-- Even if filter setup is released, check logic is existed within the source, therefore, modification of
source in the controller related to the corresponding Request Mapping(***.do) is required when the
restriction is changed. -->
<bean id="selectAnnotaionMapper"
class="egovframework.rte.ptl.mvc.handler.SimpleUrlAnnotationHandlerMapping"
p:order="1">
<property name="interceptors">
<list>
<ref local="authenticInterceptor" />
</list>
</property>
<property name="urls">
<set>
<!-- Bulletin board generation, template generation can be used after login. These can be used by
administrator only.     -->
<value>/cop/com/*.do</value>
<value>/cop/bbs/*Master*.do</value>
```

```
<value>/uat/uia/*.do</value>
</set>
</property>
</bean>

<bean id="authenticInterceptor" class="egovframework.com.cmm.interceptor.AuthenticInterceptor" >
        <!-- Access allowed URL list of login check interceptor-->
<property name="permittedURL">
<set>
<value>/uat/uia/actionLogin.do</value>
<value>/uat/uia/egovLoginUsr.do</value>
</set>
        </property>
</bean>
```

datasource related setup

- Main setup information of globals.properties (This is applied as the setup information of datasource.xml.)

Types of databse, databsedirver, databse account/password, etc. are set up.
This is identically applied to all 3 types of template, and in the case of home page, the setup in which Hsql DB can be used is additionally included.

```
  # DB server type (mysql,oracle,altibase,tibero) - Used for datasource and sqlMap file designation
Globals.DbType = mysql
  # DB user account
Globals.UserName=sht
  # DB user password
Globals.Password=sht01
  # driver information l
Globals.DriverClassName=net.sf.log4jdbc.DriverSpy
  # DB connection information
Globals.Url=jdbc:mysql://192.168.200.24:1623/SHT
```